**Name – Rohan Sasne TY-CS-D Batch-1 Roll no -02**

***---------------------------------------------------------------------------***

**Assignment-1 - B:**Tic-tac-toe game with AI approach**.**

**import java.util.Scanner;**

**public class TicTacToe {**

**private static final int *BOARD\_SIZE* = 3;**

**private static final char *PLAYER\_X* = 'X';**

**private static final char *PLAYER\_O* = 'O';**

**private static final char *EMPTY\_CELL* = ' ';**

**private static char[][] *board* = new char[*BOARD\_SIZE*][*BOARD\_SIZE*];**

**public static void main(String[] args) {**

**System.*out*.println("Welcome to our Tic-Tac-Toe Game");**

**System.*out*.println(" Symbol - X would represent Human player i.e. you");**

**System.*out*.println(" Symbol - O would represent Ai Player");**

**System.*out*.println("Enter a number from 1 to 9 to make a move.");**

***initializeBoard*();**

***playGame*();**

**}**

**private static void initializeBoard() {**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

***board*[i][j] = *EMPTY\_CELL*;**

**}**

**}**

**}**

**private static void playGame() {**

**char currentPlayer = *PLAYER\_X*;**

**boolean isGameOver = false;**

**while (!isGameOver) {**

***displayBoard*();**

**if (currentPlayer == *PLAYER\_X*) {**

***playerMove*(currentPlayer);**

**} else {**

***computerMove*(currentPlayer);**

**}**

**if (*checkWin*(currentPlayer)) {**

***displayBoard*();**

**if(currentPlayer == *PLAYER\_X*){**

**System.*out*.println("Good job human, you won!!! \uD83E\uDD73 \uD83E\uDD73");**

**}**

**if(currentPlayer == *PLAYER\_O*){**

**System.*out*.println("Opps! Looks like the AI won :) Better luck next time");**

**}**

**isGameOver = true;**

**} else if (*isBoardFull*()) {**

***displayBoard*();**

**System.*out*.println("It's a draw!");**

**isGameOver = true;**

**}**

**currentPlayer = (currentPlayer == *PLAYER\_X*) ? *PLAYER\_O* : *PLAYER\_X*;**

**}**

**}**

**private static void playerMove(char player) {**

**Scanner scanner = new Scanner(System.*in*);**

**int position;**

**do {**

**System.*out*.print("Player " + player + ", enter a position (1-9): ");**

**position = scanner.nextInt();**

**} while (!*isValidMove*(position));**

**int row = (position - 1) / *BOARD\_SIZE*;**

**int col = (position - 1) % *BOARD\_SIZE*;**

***board*[row][col] = player;**

**}**

**private static void computerMove(char player) {**

**int[] bestMove = *getBestMove*();**

***board*[bestMove[0]][bestMove[1]] = player;**

**}**

**private static int[] getBestMove() {**

**int[] bestMove = new int[]{-1, -1};**

**int bestScore = Integer.*MIN\_VALUE*;**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

**if (*board*[i][j] == *EMPTY\_CELL*) {**

***board*[i][j] = *PLAYER\_O*;**

**int score = *minimax*(*board*, 0, false);**

***board*[i][j] = *EMPTY\_CELL*;**

**if (score > bestScore) {**

**bestScore = score;**

**bestMove[0] = i;**

**bestMove[1] = j;**

**}**

**}**

**}**

**}**

**return bestMove;**

**}**

**private static int minimax(char[][] board, int depth, boolean isMaximizing) {**

**if (*checkWin*(*PLAYER\_X*)) {**

**return -10 + depth;**

**} else if (*checkWin*(*PLAYER\_O*)) {**

**return 10 - depth;**

**} else if (*isBoardFull*()) {**

**return 0;**

**}**

**if (isMaximizing) {**

**int bestScore = Integer.*MIN\_VALUE*;**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

**if (board[i][j] == *EMPTY\_CELL*) {**

**board[i][j] = *PLAYER\_O*;**

**int score = *minimax*(board, depth + 1, false);**

**board[i][j] = *EMPTY\_CELL*;**

**bestScore = Math.*max*(bestScore, score);**

**}**

**}**

**}**

**return bestScore;**

**} else {**

**int bestScore = Integer.*MAX\_VALUE*;**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

**if (board[i][j] == *EMPTY\_CELL*) {**

**board[i][j] = *PLAYER\_X*;**

**int score = *minimax*(board, depth + 1, true);**

**board[i][j] = *EMPTY\_CELL*;**

**bestScore = Math.*min*(bestScore, score);**

**}**

**}**

**}**

**return bestScore;**

**}**

**}**

**private static boolean isValidMove(int position) {**

**int row = (position - 1) / *BOARD\_SIZE*;**

**int col = (position - 1) % *BOARD\_SIZE*;**

**if (row < 0 || row >= *BOARD\_SIZE* || col < 0 || col >= *BOARD\_SIZE*) {**

**return false;**

**}**

**return *board*[row][col] == *EMPTY\_CELL*;**

**}**

**private static boolean checkWin(char player) {**

**// Check rows and columns**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**if ((*board*[i][0] == player && *board*[i][1] == player && *board*[i][2] == player)**

**|| (*board*[0][i] == player && *board*[1][i] == player && *board*[2][i] == player)) {**

**return true;**

**}**

**}**

**// Check diagonals**

**if ((*board*[0][0] == player && *board*[1][1] == player && *board*[2][2] == player)**

**|| (*board*[0][2] == player && *board*[1][1] == player && *board*[2][0] == player)) {**

**return true;**

**}**

**return false;**

**}**

**private static boolean isBoardFull() {**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

**if (*board*[i][j] == *EMPTY\_CELL*) {**

**return false;**

**}**

**}**

**}**

**return true;**

**}**

**private static void displayBoard() {**

**System.*out*.println("-------------");**

**for (int i = 0; i < *BOARD\_SIZE*; i++) {**

**for (int j = 0; j < *BOARD\_SIZE*; j++) {**

**System.*out*.print("| " + *board*[i][j] + " ");**

**}**

**System.*out*.println("|");**

**System.*out*.println("-------------");**

**}**

**System.*out*.println();**

**}**

**}**